EE440 - Introduction to Digital Imaging Systems, Fall 2018
Assignment 1

Due: 2:30 pm Wednesday 10/10/2018, before the lecture begins.

Note: Please follow the homework submission guidelines on the class webpage.

1) Image processing and commercial tools **(15 points)**

   a. Capture a digital color image of yourself and enlarge it by a factor of 2.5 in both horizontal and vertical dimensions using an image editing tool. Put the original and enlarged images in your report.
   b. Adjust **1_1.jpg** (e.g. brightness, contrast …) until you find it most pleasant. Put the adjusted image in your report.

   HINT: You could use any software you have (e.g. Adobe Photoshop, Paint/Photos in Windows).

2) MATLAB basics: image I/O and data types. **(15 points)**
   a. Load the Lena image **1_4.bmp**, using imread(), and show it using imshow().
   b. Get the type of the loaded image data (Use MATLAB function class()), and get the maximum and the minimum data values for this image (Use MATLAB function max() and min()).
   c. Convert the data to the "double" type (use MATLAB function double()), can you show the double-typed image using imshow() ?
   d. If not, given an image which has been converted to the "double" type, how do you show the image?

   HINT: MATLAB has an image value range for the default uint8 type in [0 255]. For imshow(), if the data type is double, it should be in the range [0 1]. Double type data can be converted to uint8 data, or data can be normalized to be in [0 1] for imshow().

3) Matlab basics: Matlab commands. **(20 points)**

   Write a Matlab script to do the following.

   a. Read **1_2.tif** and its associated colormap into variables named "X" and "map". Use X and "map" to convert the image to a 256-level grayscale image Y.
   b. Rotate Y 45 degrees clockwise to generate image Z.
   c. Submit images Y and Z, and the script you wrote.

   HINT: Use the Matlab commands: [X,map]=imread('1_2.tif', 'tif') , imshow(X, map) , ind2gray, imrotate.

Note: Write your own Matlab codes for the following problems.

4) Image Resolution. **(50 points)**

   a. Reduce the resolution of **1_3.asc** by a factor of 4 in both horizontal and vertical

dimensions (e.g., if the original image is 400 by 400, then result shall be 100 by 100) to create a decimated image using two different methods:

HINT: To read in an ".asc", use X=load('**1_3.asc**'). For the double type, 'imshow' only works for images with values between 0 and 1. To display the .asc image, you may use imshow(X/256).

i. Keep one pixel out of every 4x4 pixel area. Display the resulting image Y1. ***HINT:*** This can be done with only one line of code. You do not need to use for loops to accomplish this. Consider what the command A=B(1:3:20, 1:3:30) does to an image B.
ii. Replace every 4x4 pixel area in **1_3.asc** by the average value of the pixel values in that region. Display the resulting image Y2.

b. Enlarge Image Y1 by a factor of 4 in both horizontal and vertical dimensions (e.g., from 100 by 100 to 400 by 400) using:

i. Pixel repeating. (Since each pixel is blown up to a 4x4 block, the image looks "blocky".)
ii. Bilinear interpolation (do not use *interp2, use your own code*).
iii. Keep the resulting images from (b.i) and (b.ii) the same size as **1_3.asc** and compare the images.

HINT: Use $A=a$ to generate a matrix A with all entries equal to a.